Formatting your own data for the PNNL Toolbox

The PNNL_Chemometric_Methods.mlx script for MATLAB allows you to analyze optical spectra and the associated quantitative parameters using Beer's law, CLS, PCR, and PLS with the provided dataset of napalm samples. Here, you will import your own data so that it can easily be chemometrically modeled and plotted.

You will import and format your data, and give it the name `toolbox_user_data.mat`. It is recommended that you do not change any variable names; the purpose of this script is to input your own data into the variables `A_train`, `A_unknown`, `C_train`, `C_validation`, ConstituentNames, `ConcentrationUnits`, `WavenumberLabel`, and `Wavenumbers`.

NOTE: Run this script in a folder which contains your data files and this script. Alternatively, make sure to "add to path" all the folders containing the script and your target data (see MathWorks documentation for `addpath`).

```
clearvars
```

Read carefully!

Whether or not your data runs in the script is largely dependent on whether or not you format it according to the principles outlined in our manuscript. These principles are reiterated in the next section. There is very little for you to do in this script, so even if you are new to MATLAB, you should be able to follow along. Do not fear the code!

Using your data in the PNNL Toolbox

First, let's get our vocabulary on the same page.

Basics

The PNNL Chemometrics Toolbox uses the following terminology:

- *m* — the number of wavenumbers/wavelengths/frequencies (pixels) reported by your spectrometer.
- *n* — the number of constituents in your samples; in other words, the number of species in your system which produce a measurable response in that system and, in the case of the training (calibration) set, the species for which you have quantitative values.
- *p* — the number of spectra in your training or validation dataset. Generally, we assume you have one spectrum per sample (averaging several spectra into one spectrum reduces random noise!).
- A matrix — the matrix containing your spectra in row format (one spectrum per row).
- C matrix — the matrix containing your concentration data, or other quantitative data which corresponds to parameters that produce a measureable signal in the spectra, in row format (one sample per row).
- Training set — a set of data which is used to calibrate your measurement technique; usually composed of standards or samples with precisely known concentrations. You MUST have both an A and C matrix for the training set, or else you cannot perform calibration!
- Validation set — a set of data which is used to test the measurement error of your chosen technique; usually composed of standards or samples with precisely known concentrations, but different

concentrations from the training set. You MUST have both an A and C matrix in order to truly have a validation set.
- Unknown set — a set of spectra for samples with unknown concentrations of your target species (constituents). You do not have a C matrix that accompanies the spectra.

The PNNL Chemometrics Toolbox assumes that your data is organized in row format. This means that your A matrices have one spectrum per row, and a number of columns equal to the frequencies you are trying to model; i.e. your A matrices should be *p* x *m*. Your C matrices have one sample per row, and a number of columns equal to the optically active constituents which you are trying to measure; i.e. your C matrices should be *p* x *n*.

NOTE: The number of rows in your training set A and C matrices MUST be the same. The number of rows in your validation set A and C matrices MUST be the same.

Let's Start!

What Type of Data did You Collect?

In order to perform CLS, PCR, and PLS, you must at least have training spectra and concentrations (A_train and C_train).

What other data do you have?

- If you have a second set of spectra (A_unknown), write `true` for the unknownExists variablein line 2. If you do not have this file, write `false`.
- If you have the concentrations for this second set of spectra (C_validation), write `true` for the validateExists variable in line 3. If you do not have this file, write `false`.

```
unknownExists = true;  % Do you have a file containing the A_unknown spectral data?
validateExists = true; % Do you have a file containing the C_validation concentration
```

My Data is in MATLAB Format

If your data is already in MATLAB, great! Assemble the following matrices using your data:

- `Wavenumbers` — a 1 x *m* matrix containing the spectral axis for your spectra. If some of your spectra use a different axis, you will need to perform data alignment, as only one axis can be used in this script.
- `WavenumberLabel` — a string with the proper label and units for your spectral axis. E.g. `'Wavelength (nm)'`
- ConstituentNames — a cell of *n* strings containing the names of the chemical constituents in the same order as the columns of C_train.
- `A_train` — a p_{train} x *m* matrix containing the training set spectra, where *p* is the number of spectra you've taken of your training set.

- `C_train` — a p_{train} x *n* matrix containing the concentrations of each of your measured consituents in the training set, where *p* is the same as in A_train. All concentrations should be in the same units for the purpose of plot labels (if this is not possible, then you may have to edit some plot labels by hand).
- `A_unknown` — a p_{unknown} x *m* matrix containing spectra measured from samples that are not in your training set, where p_{unknown} is the number of spectra taken for this second set.
- `C_validation` — a p_{unknown} x *n* matrix containing the concentrations of each of your measured constituents in the second set of data (A_unknown), now called the validation set. Here, p_{unknown} must be the same as in `A_unknown`, and *n* must be the same as in `C_train`. Furthermore, the order of columns in `C_validation` must match the order of columns in `C_train` (e.g. column 1 is compound A and column 2 is compound B in **both** `C_train` and `C_validation`). If you do not have this data, then you have an "unknown" sample set.

My Data is Not in MATLAB Format

Do not despair. This following section will assume that you have your spectral, concentration, and x-axis (wavelength, wavenumber, frequency, etc.) in one of the following formats (see the documentation for readmatrix).

- `.txt`, `.dat`, or `.csv` for delimited text files
- `.xls`, `.xlsb`, `.xlsx`, `.xltm`, `.xltx`, or `.ods` for spreadsheet files

Importing X-axis data

Save the x-axis associated with the data in an .xlsx file on the first sheet of the document.

Insert your file's name that contains the x-axis data in the following line, replacing `myAxis.xlsx`, then run the code.

```
Wavenumbers = readmatrix('myAxis.xlsx');
```

Naming X-axis data

What is the proper label for your x-axis units? Input it in place of Units (a.u.) in the following line. This is how the x-axis will be labeled in plots.

```
WavenumberLabel = 'Units (a.u.)';
% Example from PNNL napalm data:
% WavenumberLabel='Wavenumber (cm^{-1})';
```

Importing training spectral data

Save the spectral data associated with your training set in an .xlsx file on the first sheet of the document. You should have one spectrum per row.

Insert your file's name in the following line, replacing `myAtrain.xlsx`, then run the code to import your file as a MATLAB matrix.

```
A_train = readmatrix('myAtrain.xlsx');
```

Importing training concentration data

Save the concentration data associated with your training set in an .xlsx file on the first sheet of the document. You should have one set of concentrations per row, with each column being a different chemical constituent in the samples. The order of the samples must be the same as in `A_train`.

Insert your file's name in the following line, replacing `myCtrain.xlsx`, then run the code.

```
C_train = readmatrix('myCtrain.xlsx');
```

Units of your concentration data

What are the proper units for your concentration data? Input it in place of a.u. in the following line. This is how the data will be labeled in plots.

```
ConcentrationUnits = 'a.u.';
% Example from PNNL napalm data:
% ConcentrationUnits = 'wt %';
```

Importing validation/unknown spectral data

Save the spectral data associated with your validation or unknown set in an .xlsx file on the first sheet of the document. You should have one spectrum per row.

Insert your file's name in the following line, replacing `myAunknown.xlsx`. then run the code.

```
if unknownExists
    A_unknown = readmatrix('myAunknown.xlsx');
end
```

Importing validation concentration data

Save the concentration data associated with your validation set in an .xlsx file on the first sheet of the document. You should have one set of concentrations per row, with each column being a different chemical constituent in the samples. The order of the samples must be the same as in `A_unknown`.

Insert your file's name in the following line, replacing `myCvalidation.xlsx`.

```
if validateExists
    C_validation = readmatrix('myCvalidation.xlsx');
end
```

What are your target constituents?

What are your target constituents? Input them in place of A, B, and C in the following line as you wish for them to appear in plots. If you have more than three constituents, add additional entries; if you have less than three constituents, remove entries.

```
ConstituentNames={'A', 'B', 'C'};
```

```
% Example from PNNL napalm data:
% ConstituentNames={'Benzene','Polystyrene','Gasoline'};
```

Check

This section will do some automatic checks about your data formatting. Pay attention to any output generated in this section, and take the indicated actions.

Usually you will have fewer samples (rows) than recorded frequencies (columns) in your spectral matrices.

```
if size(A_train,1)>size(A_train,2)
    disp('ARE YOU SURE THAT YOUR SPECTRA ARE IN ROW FORMAT?')
end
```

Verify the X-axis variable is a vector.

```
assert(isvector(Wavenumbers),'The X-Axis variable must be a vector.')
```

Does the number of rows in A_train match the number of rows in C_train? If no, one of these matrices has been incorrectly assembled; go back to your data and remedy the error.

```
assert(size(A_train,1)==size(C_train,1),'Your A_train and C_train matrices must have t
```

Does the number of columns in C_train match the number of columns in C_validation? If not, go back to your data and make sure these two matrices match in the *n* dimension.

```
if validateExists
    assert(size(C_train,2)==size(C_validation,2),'Your C_train and C_validation matric
end
```

Saving your data for use in the PNNL and Me Chemometrics Toolbox

Now that we have your data imported and formatting, we will save it under the name which is recognized by the many functions in the PNNL chemometrics toolbox.

```
clear unknownExists validateExists
save pnnl_chemometric_user_data.mat
```

Now that your data is imported, you should be able to input it into the `PNNL_and_Me_chemometric_toolbox.mlx` script to analyze it with CLS, PCR, and PLS.

Disclaimer

PACIFIC NORTHWEST NATIONAL LABORATORY

operated by

BATTELLE

for the

UNITED STATES DEPARTMENT OF ENERGY

under Contract DE-AC05-76RL01830